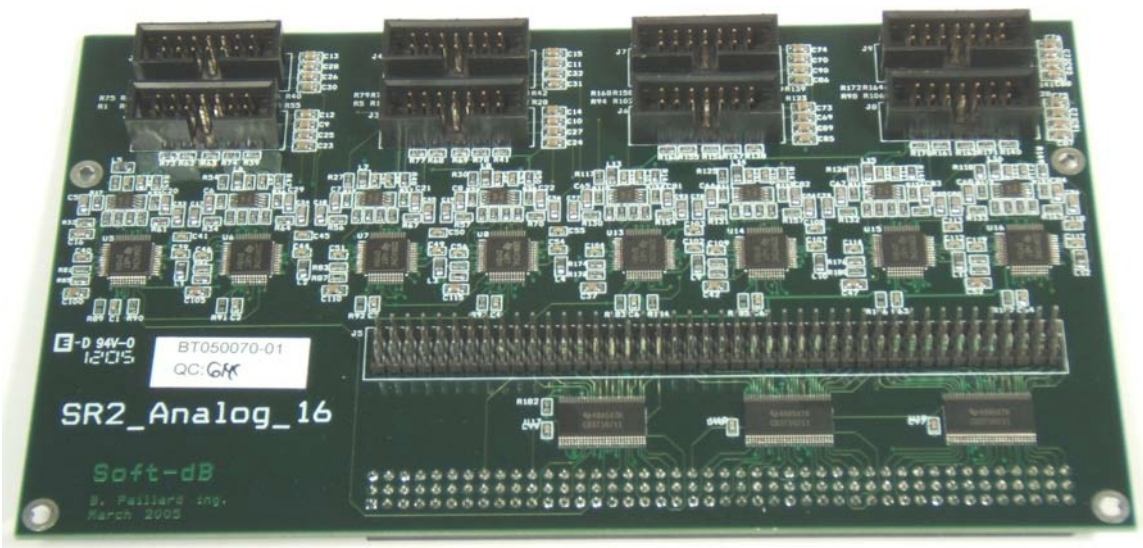
SR2_Analog_16
Carte d'E/S analogiques

Manuel de l'utilisateur



Bruno Paillard
Rev 03 6 juillet 2005

Caractéristiques principales

SR2_Analog_16 est une carte additionnelle pour la carte *Signal_Ranger_mk2.* Elle offre deux fonctions :

- Fournit 16 entrées et 16 sorties analogiques.
- Fournit des commutateurs de bus sur les E/S FPGA de *Signal_Ranger_mk2*. Ces commutateurs de bus assurent la protection des E/S et des circuits externes qui y sont connectés au cas où *Signal_Ranger_mk2* serait alimenté alors que les circuits externes ne le seraient pas, ou vice versa. En outre, les commutateurs de bus rendent les E/S configurées en tant qu'entrées tolérantes au 5V.

Trois des E/S FPGA du *Signal_Ranger_mk2* sont utilisées pour gérer les circuits d'interface analogique (AIC). Par conséquent, ces E/S ne sont pas disponibles pour l'utilisateur lorsque *SR2_Analog_16* est présent. De plus, les AICs utilisent aussi complètement les deux McBSPs du DSP. Ils ne sont donc pas non plus mis à la disposition de l'utilisateur.

Données techniques :

- **Entrées analogiques différentielles**
  - Nombre d'entrées : 16
  - Résolution : 16 bits
  - Bruit : 84dB
  - Taux d'échantillonnage : 0 à 26kHz
  - Largeur de bande : 0 à Fs/2 (y compris DC)
  - Gamme dynamique : Simple effet : +-10V - Différentielle : +-20V
  - Impédance d'entrée : 85 kΩ
  - Gain réglable : 0 à 42dB par pas de 1,5dB, 48dB, 54dB
  - Filtre anti-crénelage :
    - IIR : Retard de groupe 5 échantillons
    - FIR : Retard de groupe 18 échantillons
- **Entrées microphones**
  - L'entrée microphone peut être sélectionnée sur n'importe laquelle des 16 entrées analogiques.
  - Interface directe avec un microphone électret (charge de 10 kΩ).
  - Filtre passe-haut coupé à 16 Hz
- **Sorties analogiques à une extrémité**
  - Nombre de sorties : 16
  - Résolution : 16 bits
  - Bruit : 90dB
  - Taux d'échantillonnage : 0 à 26kHz
  - Largeur de bande : 0 à Fs/2 (y compris DC)
  - Gamme dynamique : +-2V
  - Impédance de sortie : 50 Ω
  - Capacité Source/Sink : 1mA
  - Atténuation réglable : 0 à -42dB par pas de 1,5dB, -48dB, -54dB
  - Filtre anti-crénelage :
    - IIR : Retard de groupe 5 échantillons
    - FIR : Retard de groupe 18 échantillons

Logiciel :

Un pilote d'E/S analogiques et un code d'exemple sont fournis.

Installation

Connexion à *Signal_Ranger_mk2*

Lorsque la carte *SR2_Analog_16* est achetée séparément de la carte *Signal_Ranger_mk2*, le connecteur J10 de la *carte SR2_Analog_16* doit être relié au connecteur J4 de la *carte Signal_Ranger_mk2. Signal_Ranger_mk2* fournit l'alimentation et les signaux d'interface à *SR2_Analog_16*. *SR2_Analog_16* fournit un connecteur d'extension J5, qui fournit les mêmes signaux FPGA présents sur le connecteur J4 de *Signal_Ranger_mk2*, à l'exception des signaux utilisés pour l'interface des Circuits d'Interface Analogique.

Remarque : Lorsque SR2_Analog_16 est associé à Signal_Ranger_mk2, l'utilisateur ne doit pas s'alimenter à partir du connecteur d'extension J5, à l'exception du connecteur +5V.

Configuration du FPGA

Pour pouvoir gérer les circuits d'interface analogique, le FPGA doit être configuré différemment de sa configuration d'usine par défaut *Signal_Ranger_mk2.* Cette configuration spéciale est déjà placée dans la Flash du *Signal_Ranger_mk2* lorsque les deux cartes sont achetées ensemble. Cependant, lorsque la *carte SR2_Analog_16* est achetée séparément, le fichier *SR2_Analog_16.rbt* doit être programmé dans la Flash pour fournir la fonctionnalité requise à la mise sous tension. L'utilisateur peut également choisir de charger dynamiquement ce fichier de configuration FPGA avant de charger tout code DSP qui doit piloter les AIC. Les détails de ce fichier d'implémentation logique sont donnés dans le document *SR2_Analog_FPGA_Logic.pdf.* Les principales différences entre cette logique et la logique FPGA *Signal_Ranger_mk2* par défaut sont les suivantes :

- Le port_D a un nombre d'entrées-sorties réduit à 12 bits.
- Les 3 bits IO qui ont été prélevés sur le port_D sont maintenant utilisés pour piloter les signaux suivants des AIC :
  - *AIC_Clk_Out :* Conduit le signal d'horloge maître de l'AIC
  - *AIC_PwrDwn :* Pilote le signal de mise hors tension de l'AIC
  - *AIC_Reset :* Pilote le signal de réinitialisation de l'AIC
- Un nouveau registre AIC_Control est défini à l'adresse d'octet $C00022_H$. Les deux bits inférieurs de ce registre contrôlent les lignes *AIC_PwrDwn* et *AIC_Reset.*

Remarque : Les applications d'auto-test et de démonstration présentées ci-dessous imposent un chargement dynamique de la logique du FPGA avec le fichier SR2_Analog_16.rbt. Ces applications ne s'appuient pas sur le fichier de configuration qui peut être chargé dans le FPGA à la mise sous tension à partir de la Flash ROM. Quelle que soit la logique présente dans le FPGA, elle est écrasée par le chargement dynamique effectué par ces applications.

Autocontrôle

Une application d'auto-test nommée *SR2_Analog_SelfTest* est fournie pour tester la carte SR2_Analog_16. Cette application ne peut fonctionner que si la carte Signal_Ranger_mk2 est fonctionnelle. En cas de doute sur la fonctionnalité de la carte Signal_Ranger_mk2, l'utilisateur doit d'abord la tester à l'aide de l'application *SR2_SelfTest.*
Le panneau avant de l'application *SR2_Analog_SelfTest* est illustré ci-dessous :

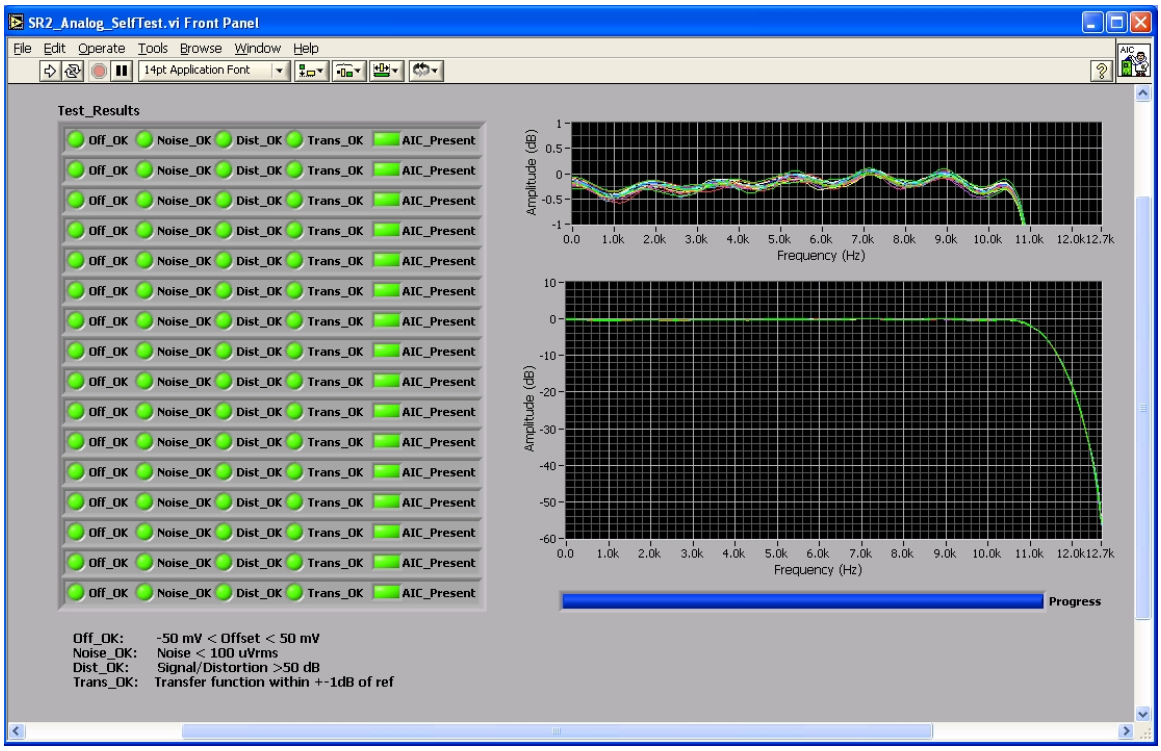


Figure 1 : Application SR2_Analog_Self_Test

Pour lancer l'application, il suffit de cliquer sur la flèche en haut à gauche de la fenêtre.
L'application charge la logique FPGA, puis effectue les tests suivants :

- Dénombre le nombre d'AIC dans chaque chaîne
- Teste le décalage et le bruit des AICs
- Teste la fonction de transfert entrée/sortie (bouclage analogique) des AIC, ainsi que leur distorsion.

Les tests de fonction de transfert et de distorsion sont effectués en utilisant le mode de bouclage analogique de l'AIC. Ils ne testent donc pas l'amplificateur opérationnel de sortie, ni les connecteurs. Cependant, le signal de test est toujours émis sur les connecteurs.
Les tests de fonction de transfert et de distorsion sont effectués à l'aide d'un signal blanc de pleine amplitude. Ce signal est susceptible de produire beaucoup de bruit si les sorties sont connectées à d'autres équipements, tels qu'un amplificateur de puissance et un haut-parleur.

Indicateurs :

Test_Results : Ce tableau indique le résultat du test suivant pour chaque canal. Le canal 0 est en haut, le canal 15 est en bas.

- Off_OK Indique que le décalage d'entrée est compris entre +-50 mV. Remarque : l'AIC est en boucle interne pendant ce test, ce qui indique également le décalage de sortie.
- Bruit_OK Indique que le bruit se situe dans les 100 µVrms.
- Dist_OK Indique que le rapport signal sur (bruit+distorsion) est d'au moins 50 dB.
- Trans_OK Indique que la fonction de transfert entrée/sortie (bouclage) se situe à +-1 dB d'une fonction de transfert de référence. Ce test n'est effectué que sur 90 % de la largeur de bande, car des erreurs de plus de 2 dB dans la bande de transition ne sont pas inhabituelles.

- **Amplitude (dB)** : Trace les fonctions de transfert entrée/sortie pour chaque AIC. Le graphique supérieur est un zoom sur la partie passe-bande du graphique.
- **Progrès** : Indique l'état d'avancement des tests de distorsion et de fonction de transfert.

Configuration du FPGA et du DSP

Configuration du FPGA

Avant d'essayer de gérer les AIC, le FPGA doit être configuré avec la logique *SR2_Analog_16.rbt*. Comme indiqué ci-dessus, configuration peut être placée dans la mémoire Flash de façon à être chargée dans le FPGA à la mise sous tension. Alternativement, la configuration peut être chargée dynamiquement à partir de l'application PC, ou directement à partir du code DSP, avant que les fonctions du pilote DSP AIC ne soient invoquées. Le code d'exemple utilise le chargement dynamique. Il ne dépend donc pas de l'état du FPGA immédiatement après la mise sous tension.

Certains utilisateurs peuvent avoir besoin de placer une logique supplémentaire dans le FPGA. Dans ce cas, les exigences minimales de la logique FPGA pour supporter correctement le pilote DSP AIC fourni sont indiquées ci-dessous :

- Il doit y avoir un registre *AIC_Control* à l'adresse d'octet $C00022_H$dans le FPGA. Les deux bits inférieurs de ce registre doivent contrôler les lignes de sortie *AIC_PwrDwn* et *AIC_Reset*, comme décrit dans *SR2_Analog_FPGA_Logic.pdf*.
- Une sortie d'horloge DSP doit être utilisée pour générer le signal d'horloge maître de l'AIC. La seule contrainte est que le signal d'horloge maître de l'AIC doit être inférieur à 100MHz, et doit sortir sur la broche 11 du FPGA pour être correctement acheminé vers les AICs (voir le brochage du projet FPGA par défaut). Dans la configuration par défaut, la sortie DSP *CLKO* est utilisée pour générer un signal de 150MHz sur la broche 52 du FPGA. Ce signal est divisé par 2 dans le FPGA et acheminé vers la broche 11 du FPGA pour générer un signal d'horloge de 75MHz vers les entrées de l'horloge maîtresse de l'AIC. En plus d'utiliser la sortie DSP *ClkOut* pour générer le signal d'horloge maître de l'AIC, l'utilisateur a également la possibilité d'utiliser les sorties DSP *TIM0* ou *TIM1*. Ces sorties d'horloge DSP alternatives sont acheminées vers les broches 128 et 127 du FPGA respectivement.
- Les sorties FPGA *AIC_PwrDwn* et *AIC_Reset* doivent être synchronisées avec la sortie FPGA *AIC_Clk_Out* comme expliqué dans *SR2_Analog_FPGA_Logic.pdf*. Bien que ce soit un bon début, il n'est pas suffisant de copier la partie de la logique relative à la gestion de l'AIC à partir du projet Xilinx ISE fourni. Les timings décrits dans *SR2_Analog_FPGA_Logic.pdf* doivent être vérifiés.
- Le brochage du FPGA pour *AIC_Clk_Out*, *AIC_PwrDwn* et *AIC_Reset* doit être conservé, car ils sont liés à la disposition de la carte *SR2_Analog_16*.

Configuration de la minuterie DSP

Dans la configuration par défaut, le signal *AIC_Clk_Out* fourni par le FPGA aux entrées de l'horloge maîtresse de l'AIC provient de la sortie *CLKOut* du DSP. Le noyau de mise sous tension et le noyau de téléchargement de l'hôte configurent tous deux la sortie *CLKOut* pour un signal de sortie de 150 MHz. Si la configuration de la sortie *CLKOut* n'est pas modifiée par le code utilisateur du DSP, cela fournit un signal *AIC_Clk_Out de* 75MHz aux entrées de l'horloge maîtresse de l'AIC immédiatement après la configuration du FPGA.

Description du matériel

Carte des connecteurs

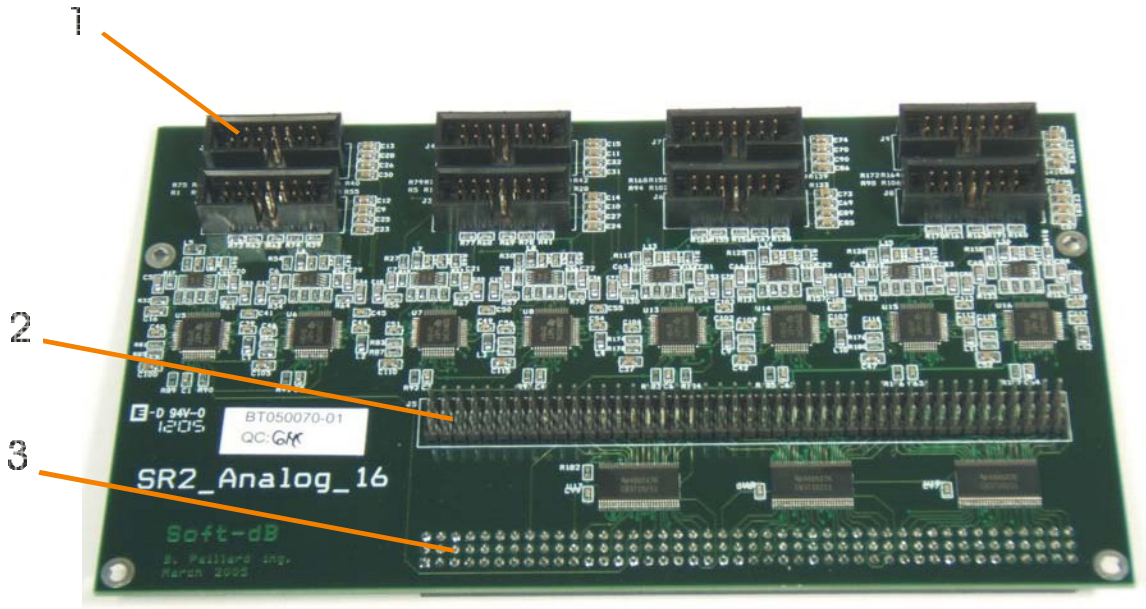
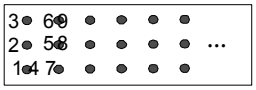


Figure 2

Légende :

1	Connecteurs analogiques J1, J2, J3, J4, J6, J7, J8 et J9
2	Connecteur d'extension J5
3	Signal_Ranger_mk2 connecteur J10

Connecteur d'extension J5



Non	Fonction	Non	Fonction	Non	Fonction
1	+5V	2	Gnd	3	+2.5V
4	+1.8V	5	Gnd	6	+1.2V
7	+1.26V	8	Gnd	9	+3.3V
10	-3.3V	11	Gnd	12	DSP_Reset
13	NC	14	Gnd	15	NC
16	NC	17	Gnd	18	NC
19	NC	20	Gnd	21	NC
22	NC	23	Gnd	24	NC
25	NC	26	Gnd	27	NC
28	NC	29	Gnd	30	NC
31	UART_Rx	32	Gnd	33	UART_Tx
34	NC	35	Gnd	36	Présence

37	FPGA_17	38	Gnd	39	NC
40	FPGA_14	41	Gnd	42	NC
43	FPGA_12	44	Gnd	45	NC
46	FPGA_10	47	Gnd	48	FPGA_8
49	FPGA_7	50	Gnd	51	FPGA_6
52	FPGA_5	53	Gnd	54	FPGA_4
55	FPGA_2	56	Gnd	57	FPGA_1
58	FPGA_141	59	Gnd	60	FPGA_140
61	FPGA_137	62	Gnd	63	FPGA_135
64	FPGA_68	65	Gnd	66	FPGA_69
67	FPGA_70	68	Gnd	69	FPGA_73
70	FPGA_74	71	Gnd	72	FPGA_76
73	FPGA_77	74	Gnd	75	FPGA_78
76	FPGA_79	77	Gnd	78	FPGA_80
79	FPGA_82	80	Gnd	81	FPGA_83
82	FPGA_84	83	Gnd	84	FPGA_85
85	FPGA_86	86	Gnd	87	FPGA_87
88	FPGA_89	89	Gnd	90	FPGA_90
91	FPGA_92	92	Gnd	93	FPGA_93
94	FPGA_95	95	Gnd	96	FPGA_96
97	FPGA_97	98	Gnd	99	FPGA_98
100	FPGA_99	101	Gnd	102	FPGA_100
103	FPGA_102	104	Gnd	105	FPGA_103
106	FPGA_104	107	Gnd	108	FPGA_105
109	FPGA_107	110	Gnd	111	FPGA_108
112	FPGA_112	113	Gnd	114	FPGA_113
115	FPGA_116	116	Gnd	117	FPGA_118
118	FPGA_119	119	Gnd	120	FPGA_122
121	FPGA_123	122	Gnd	123	FPGA_124
124	FPGA_125	125	Gnd	126	FPGA_129
127	FPGA_130	128	Gnd	129	FPGA_131
130	FPGA_132	131	Gnd	132	FPGA_HS_EN

Tableau 1 : Brochage du connecteur J5

Différences avec le connecteur d'expansion J4 *de Signal_Ranger_mk2*

Le connecteur d'extension J5 du *SR2_Analog_16* est disposé de manière presque identique à celui du *SR2_Analog_16*.
Le connecteur J4 du *Signal_Ranger_mk2*. Les différences sont décrites ci-dessous :

Signaux McBSP

Les signaux McBSP pour McBSP0 et McBSP1 ne sont pas fournis sur le connecteur d'extension J5. Ils sont utilisés pour gérer les AIC et ne sont donc plus disponibles pour un usage général.

FPGA_15, FPGA_13, FPGA_11

Les bits d'E/S FPGA *FPGA_11, FPGA_13, FPGA_15* ne sont pas fournis sur les broches J5 45, 42 et 39. Ils sont utilisés pour gérer les AIC et ne sont donc plus disponibles pour un usage général.

Présence

Une nouvelle entrée de *présence* est fournie sur la broche 36 de J5, qui n'existait pas sur le connecteur J4 de *Signal_Ranger_mk2.* Cette entrée est utilisée pour activer les commutateurs de bus qui isolent tous les signaux de J5 de la carte *Signal_Ranger_mk2.*
Toutes les broches de J5 sont isolées par les commutateurs de bus, à l'exception des connexions d'alimentation.

Les commutateurs de bus ont deux fonctions :

- Ils isolent *Signal_Ranger_mk2* d'une carte utilisateur connectée sur J5 lorsqu'une carte est alimentée mais pas l'autre. Ceci est essentiel car, en l'absence de commutateurs, les pilotes d'une carte pourraient piloter des étages d'entrée non alimentés sur l'autre carte, ce qui pourrait les endommager.
- Ils assurent la conversion de niveau entre la carte utilisateur, qui peut piloter des niveaux jusqu'à 5V, et les entrées du *Signal_Ranger_mk2* qui ne sont pas tolérantes à 5V. En bref, les commutateurs rendent les entrées du *Signal_Ranger_mk2* tolérantes à 5V.

Pour assurer la première fonction, les interrupteurs ne doivent être activés (basse impédance) que lorsque la carte fille de l'utilisateur est alimentée. L'entrée *Présence* remplit cette fonction. Cette entrée est normalement tirée vers le haut par une résistance de 10 kΩ sur *SR2_Analog_16*. L'entrée doit être tirée vers le bas par un pilote à drain ouvert sur la carte fille de l'utilisateur lorsque la carte fille est correctement alimentée. Tirer *la Présence* vers le bas lorsque la carte fille de l'utilisateur n'est pas correctement alimentée (avant qu'elle ne soit alimentée ou après qu'elle ait été mise hors tension) expose les étages d'entrée de la carte de l'utilisateur aux dommages infligés par les broches du FPGA qui peuvent être configurées comme des sorties. Selon la façon dont le FPGA est configuré à ce moment-là, cela peut ne pas poser de problème. Par exemple, si le FPGA n'est pas configuré, ses entrées/sorties sont configurées comme des entrées et ne conduisent pas de courant. Dans ce cas, *la Présence* peut être liée en permanence à la masse. Le fait de tirer *Presence* vers le bas lorsque la carte fille de l'utilisateur est correctement alimentée mais que *Signal_Ranger_mk2* ne l'est pas N'EXPOSE PAS les entrées du FPGA aux dommages qui pourraient être infligés par les pilotes de la carte fille de l'utilisateur. En effet, les interrupteurs sont désactivés lorsque *Signal_Ranger_mk2* n'est pas alimenté.
La seconde fonction est assurée automatiquement lorsque les interrupteurs sont activés (lorsque *La présence* est faible).

Alimentations

+5V

Il s'agit de la même ligne +5V qui est amenée au connecteur d'alimentation J2. Le courant maximum qui peut être tiré de cette broche est de 500mA. Il peut être limité par la capacité de l'alimentation utilisée. C'est la seule alimentation qui peut être utilisée par l'utilisateur lorsque *le SR2_Analog_16* est couplé au *Signal_Ranger_mk2*.

+2.5V

Il s'agit de l'alimentation Vccaux du FPGA. Aucun courant significatif ne doit être tiré de cette broche.

+1.8V

Cette alimentation est utilisée pour alimenter le cœur logique des AIC. Aucun courant significatif ne doit être tiré de cette broche.

+1.2V

Il s'agit de l'alimentation Vccint du FPGA. Aucun courant significatif ne doit être tiré de cette broche.

+1.26V

Il s'agit de l'alimentation CVdd du DSP. Aucun courant significatif ne doit être tiré de cette broche.

+3.3V

Cette alimentation est utilisée par le DSP, le FPGA, la Flash, la SDRAM et le contrôleur USB. Aucun courant significatif ne doit être tiré de cette broche.

-3.3V

Cette alimentation est utilisée par les étages de sortie analogique. Aucun courant significatif ne doit être tiré de cette broche.

Broches DSP

DSP_Reset

Il s'agit de la broche de réinitialisation du DSP. Elle est activée (bas) à la mise sous tension et sous le contrôle du contrôleur USB. Elle peut être utilisée pour réinitialiser la logique externe chaque fois que le DSP est réinitialisé.

UART

Les broches UART Tx et Rx du DSP sont fournies sur J5.

Broches FPGA

FPGA_i

Toutes les E/S libres et les broches d'horloge du FPGA sont fournies sur J5.

FPGA_HS_EN

Il s'agit de la broche HSWAP_EN du FPGA. Si elle est tirée vers le haut ou laissée flottante, toutes les E/S du FPGA sont flottantes pendant la configuration. Si elle est tirée vers le bas, les E/S du FPGA sont tirées vers le haut pendant la configuration. L'état des E/S du FPGA après la configuration est défini par la logique chargée dans le FPGA, et l'état de HSWAP_EN n'a aucune incidence.

Remarque : FPGA_HS_EN est transmis au connecteur d'extension J5 par l'intermédiaire du commutateur de bus. Ainsi, pour que le FPGA voie cette broche dans un état bas pendant la configuration, les commutateurs de bus doivent être activés avant la configuration du FPGA.

Connecteurs d'E/S analogiques J1, J2, J3, J4, J6, J7, J8, J9

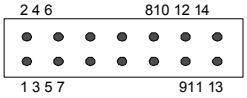
Affectation des connecteurs

Connecteur Non	McBSP/CODEC_Addresses
J1	McBSP_0 / CODEC_0 & CODEC_1 (maître)
J2	McBSP_0 / CODEC_2 & CODEC_3
J3	McBSP_0 / CODEC_4 & CODEC_5
J4	McBSP_0 / CODEC_6 & CODEC_7
J6	McBSP_1 / CODEC_0 & CODEC_1 (maître)
J7	McBSP_1 / CODEC_2 & CODEC_3

J8	McBSP_1 / CODEC_4 & CODEC_5
J9	McBSP_1 / CODEC_6 & CODEC_7

Tableau 2 : Affectation des connecteurs d'E/S analogiques

Brochage du connecteur



Non	Fonction	Non	Fonction
1	AIC_Sortie_2	2	AIC_Sortie_3
3	Gnd	4	Gnd
5	AIC_InP2	6	AIC_InP3
7	AIC_InM2	8	AIC_InM3
9	Gnd	10	Gnd
11	InP4/InM4 (Mic_In)	12	InP1/InM1 (Mic_In)
13	+Vbias	14	+Vbias

Tableau 3 : Brochage du connecteur d'E/S analogiques

Sorties simples

Deux canaux de sortie asymétriques sont fournis sur les broches 1 et 2. Ces sorties sont transformées des sorties différentielles référencées à 1,35V de l'AIC en une sortie asymétrique référencée à 0V à l'aide d'un étage d'amplification différentielle. L'amplificateur différentiel a un gain de 1, de sorte que la plage dynamique des sorties asymétriques est de +-2V. La sortie différentielle OutP2/OutM2 de l'AIC est acheminée vers la broche 1 du connecteur, tandis que la sortie OutP3/OutM3 de l'AIC est acheminée vers la broche 2. La barre transversale à l'intérieur de l'AIC doit être utilisée pour sélectionner laquelle des deux sorties CODEC est acheminée vers OutP2/OutM2 (broche 1), et laquelle est acheminée vers OutP3/OutM3 (broche 2). Le choix de toute autre sortie de l'AIC ramènera effectivement le signal à zéro sur la broche correspondante du connecteur.

Entrées AIC

Quatre des entrées de l'AIC sont amenées au connecteur. Deux d'entre elles (InP2/InM2 et InP3/InM3) sont configurées comme des entrées différentielles standard avec une impédance d'entrée de 85 kΩ et une plage dynamique de +-10V. Les deux autres (InP1/InM1 et InP4/InM4) sont configurées comme des entrées de microphone avec une dynamique de +- 2V, une polarisation de microphone et une fréquence de coupure basse de 16Hz. L'utilisateur doit utiliser la barre transversale à l'intérieur de l'AIC pour choisir quelle entrée de connecteur est acheminée vers quelle entrée CODEC.
Cela offre une grande flexibilité pour assigner n'importe quelle entrée différentielle ou microphone sur le connecteur à n'importe quel CODEC dans l'AIC.

Entrées différentielles

L'entrée différentielle InP2/InM2 de l'AIC est fournie sur les broches 5 et 7 et l'entrée différentielle InP3/InM3 de l'AIC est fournie sur les broches 6 et 8. La barre transversale de l'AIC doit être utilisée pour acheminer l'une ou l'autre des entrées différentielles vers l'un ou l'autre des CODEC de l'AIC.

Biais d'entrée pour les mesures couplées en courant continu

Sur chaque entrée différentielle, un simple réseau de résistances et la sortie VBias de l'AIC sont utilisés pour porter la plage dynamique de chaque entrée asymétrique à +-10V et pour décaler chaque entrée asymétrique de façon à ce qu'elle ne soit pas trop sensible aux variations de tension.

qu'il est référencé à 0V au niveau du connecteur, plutôt qu'à 1,35V - comme c'est le cas au niveau de l'AIC. Pour que ce réseau de résistances remplisse sa fonction, deux conditions doivent être remplies :

- Chaque entrée asymétrique de la paire différentielle (par exemple la broche 5 et la broche 7) doit être pilotée par un pilote à faible impédance référencé à 0V.
- L'AIC doit être réglé de manière à ce que sa sortie VBias soit de 2,35V au lieu de 1,35V.

Si une entrée de la paire différentielle est laissée flottante, plutôt que d'être pilotée à basse impédance, la tension statique sur l'entrée correspondante de l'AIC est de 1,5V. Comme chaque entrée de la paire différentielle de l'AIC est référencée à 1,35V, cette tension de 1,5V est équivalente à un décalage statique de 150mV sur l'entrée de l'AIC. Si les deux entrées de la paire différentielle sont laissées flottantes ou pilotées à haute impédance, le même décalage est appliqué aux deux entrées de la paire différentielle et il s'. Cependant, dans tous les , ce décalage statique réduit la plage dynamique chaque entrée asymétrique de la paire différentielle, abaissant ainsi le seuil de saturation de l'étage d'entrée différentiel de l'AIC.

Si une source asymétrique est utilisée pour piloter la paire différentielle, il suffit de la connecter à l'entrée InP ou InM, et de connecter l'autre entrée à la masse. Ajoutez 6 dB au gain d'entrée pour compenser labsence du signal différentiel.

Biais d'entrée pour les mesures couplées en courant alternatif

Si les entrées de la paire différentielle sont pilotées par un couplage CA (un condensateur), la sortie VBias de l'AIC doit être réglée sur 1,35V, plutôt que sur 2,35V. Sinon, un décalage statique équivalent à 150mV est appliqué à chaque entrée de la paire différentielle, ce qui réduit d'autant sa plage dynamique.

Note : Le réglage VBias de chaque AIC est commun aux deux CODEC de l'AIC : Le réglage du VBias pour chaque AIC est commun aux deux CODEC de l'AIC, l'utilisateur doit donc choisir le couplage AC (VBias= 1,35V) ou le couplage DC (VBias= 2,35V) pour les deux canaux de l'AIC.

Entrées microphones

Une entrée de microphone à électret peut être sélectionnée comme entrée de l'un ou l'autre CODEC de l'AIC. Pour ce faire, le microphone à électret doit être connecté entre les broches 9 et 11 (pour InP1/InM1), avec le positif sur la broche 11, et/ou entre les broches 10 et 12 (pour InP4/InM4), avec le positif sur la broche 12. La barre transversale de l'AIC doit être utilisée pour sélectionner l'entrée CODEC acheminée vers InP1/InM1 (broches 9 et 11) et celle acheminée vers InP4/InM4 (broches 10 et 12). L'AIC doit être réglé de manière à ce que son Vbias soit de 2,35 V pour polariser correctement la plupart des microphones à électret.

Description du logiciel

Application de démonstration pour le suivi des signaux

L'application Signal Tracker a été conçue pour permettre le test et l'évaluation des canaux d'entrée/sortie analogiques. L'application permet à l'utilisateur d'envoyer des signaux de test à une sortie de canal sélectionnée et de surveiller le signal échantillonné sur une entrée sélectionnée. Les entrées sont affichées à la fois termes de signaux temporels et de spectres d'énergie instantanés ou moyennés. Les spectres d'énergie moyennés sont utiles pour évaluer le bruit d'entrée. Le panneau avant de l'application est divisé en plusieurs onglets, un pour chaque groupe de fonctions.

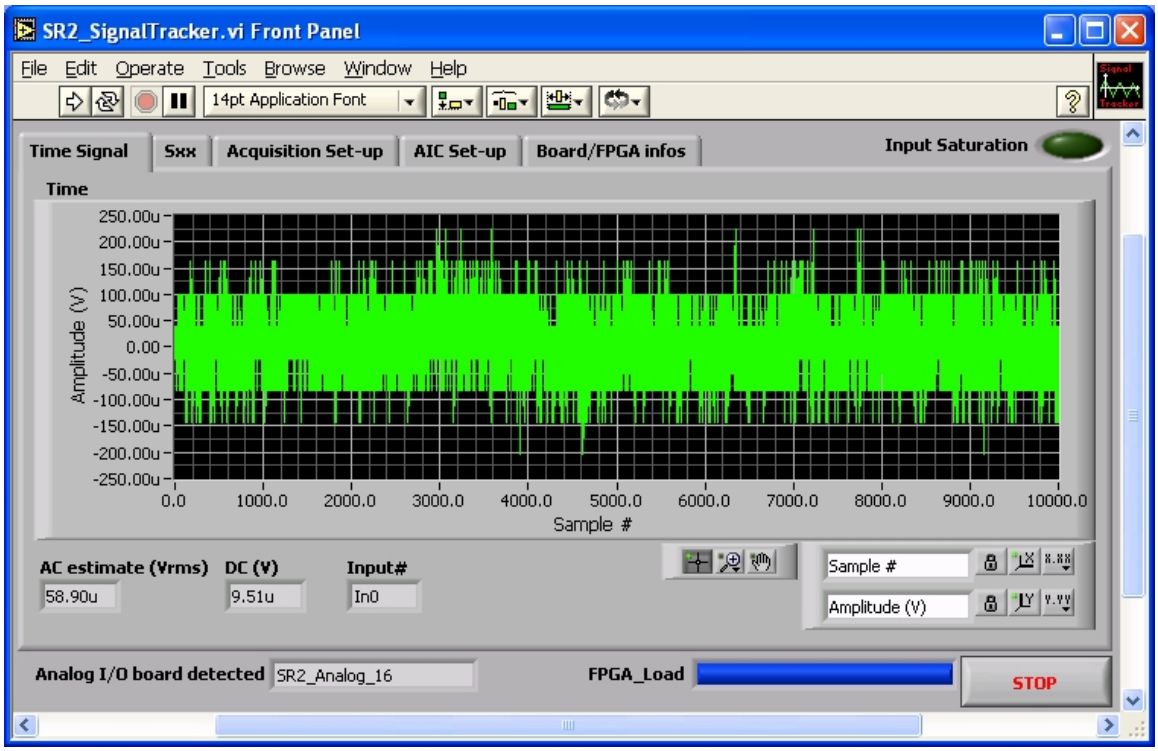


Figure 3 : Application SR2_SignalTracker - Onglet Signal temporel

Pour lancer l'application, il suffit de cliquer sur la flèche blanche en haut à gauche de la fenêtre. L'application envoie des blocs d'échantillons de la longueur et de la forme d'onde sélectionnées à la sortie sélectionnée, et enregistre des blocs d'échantillons de la même longueur sur l'entrée sélectionnée. Les échantillons d'entrée enregistrés sont synchronisés avec les échantillons de sortie.

Onglet Signal temporel

Indicateur de temps

L'onglet signal temporel présente un tracé temporel du signal échantillonné sur l'entrée sélectionnée. L'échelle d'amplitude prend en compte le gain de la chaîne analogique et le gain du PGA, de sorte que l'amplitude du signal est représentée en Volts au niveau du connecteur.

Estimation CA (Vrms) Indicateur

Cet indicateur présente la valeur efficace du signal d'entrée (tout décalage en courant continu est supprimé avant le calcul de la valeur efficace), ainsi que la valeur moyenne en courant continu.

Estimation CA (Vrms) Indicateur

Cet indicateur présente la valeur moyenne en courant continu du signal temporel.

Onglet Sxx

Indicateur de spectre

L'indicateur de *spectre* présente le spectre de puissance instantané ou moyenné du bloc échantillonné d'entrée.

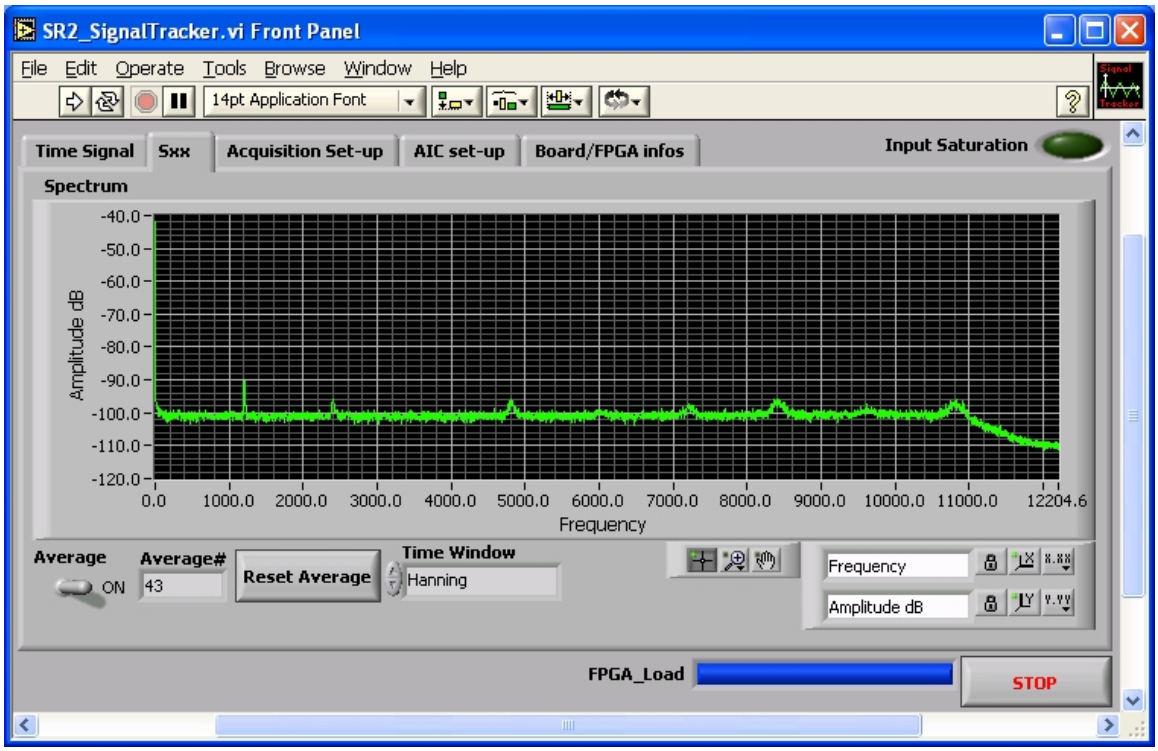


Figure 4 : Application SR2_SignalTracker - onglet Sxx

L'échelle verticale est en dB. Une valeur de 0 dB représente une amplitude de 1Vrms.

Contrôle moyen

Pour calculer la moyenne du spectre de puissance, il suffit de placer la commande *Average* en position ON.

Bouton de réinitialisation de la moyenne

Le bouton *Réinitialiser la moyenne* permet de réinitialiser la moyenne.

Sélecteur de fenêtre temporelle

Une fenêtre optionnelle peut être choisie dans la liste des *fenêtres temporelles*.

Contrôles du graphique et du zoom

Les commandes du graphique peuvent être utilisées pour modifier le facteur de zoom. Par défaut, le graphique est mis à l'échelle automatiquement en X et en Y, ce qui est indiqué par les verrous fermés à côté de chaque nom d'échelle. Pour désactiver l'échelle automatique, il suffit d'appuyer sur le bouton de verrouillage.

Onglet Configuration de l'acquisition

L'onglet Configuration de l'acquisition présente les différents contrôles pour la configuration de l'acquisition.

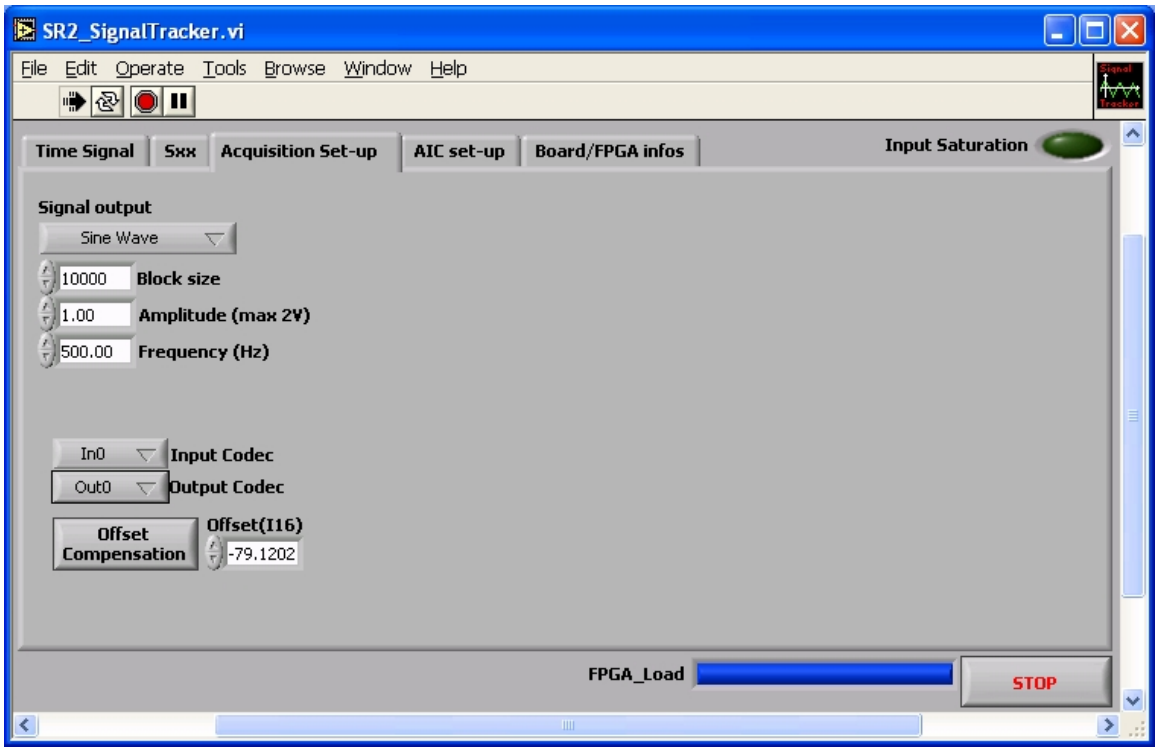


Figure 5 : **Application SR2_SignalTracker - Onglet Configuration de l'acquisition**

Contrôle de la sortie du signal

La commande *Signal output* sélectionne un type de forme d'onde à partir d'une liste de formes d'onde prédéfinies. La sélection *No Output* envoie zéro échantillon à la sortie.

Contrôle de la taille des blocs

Le contrôle de la *taille des blocs* définit le nombre d'échantillons envoyés à la sortie et enregistrés de manière synchrone à partir de l'entrée.

Contrôle de l'amplitude

La commande d'*amplitude* permet de régler l'amplitude de la forme d'onde de sortie. Les échantillons de sortie sont calculés en fonction de l'amplitude sélectionnée, ainsi que du gain de sortie sélectionné (PGA).

Contrôle de la fréquence

Le contrôle de la *fréquence* n'est utilisé que pour les formes d'onde périodiques. Elle permet de régler la fréquence fondamentale de la forme d'onde.

Contrôle du codec d'entrée

La commande *Input Codec* permet de sélectionner le canal d'entrée entre 0 et 15.

Contrôle du codec de sortie

Le *codec de sortie* sélectionne le canal de sortie entre 0 et 15.

Contrôle de la compensation du décalage

Le bouton *Compensation de décalage* permet d'effectuer une compensation de décalage. Cette procédure lit un bloc d'échantillons d'entrée à partir de l'entrée sélectionnée tout en envoyant des échantillons nuls. La moyenne des

est ensuite soustrait des échantillons d'entrée. Par conséquent, si un décalage est présent sur l'entrée sélectionnée, il est ramené à zéro. La moyenne est affichée dans l'indicateur *Offset(I16)*. Cet indicateur est mis à l'échelle en bits. La compensation du décalage est logicielle.

Offset(I16) Contrôle

L'indicateur *Offset(I16)* peut également servir de contrôle. Le simple fait de modifier le contenu de ce champ impose un décalage logiciel aux échantillons d'entrée enregistrés.

Onglet de configuration de l'AIC

L'onglet Configuration AIC présente les différents contrôles pour la configuration de l'acquisition.

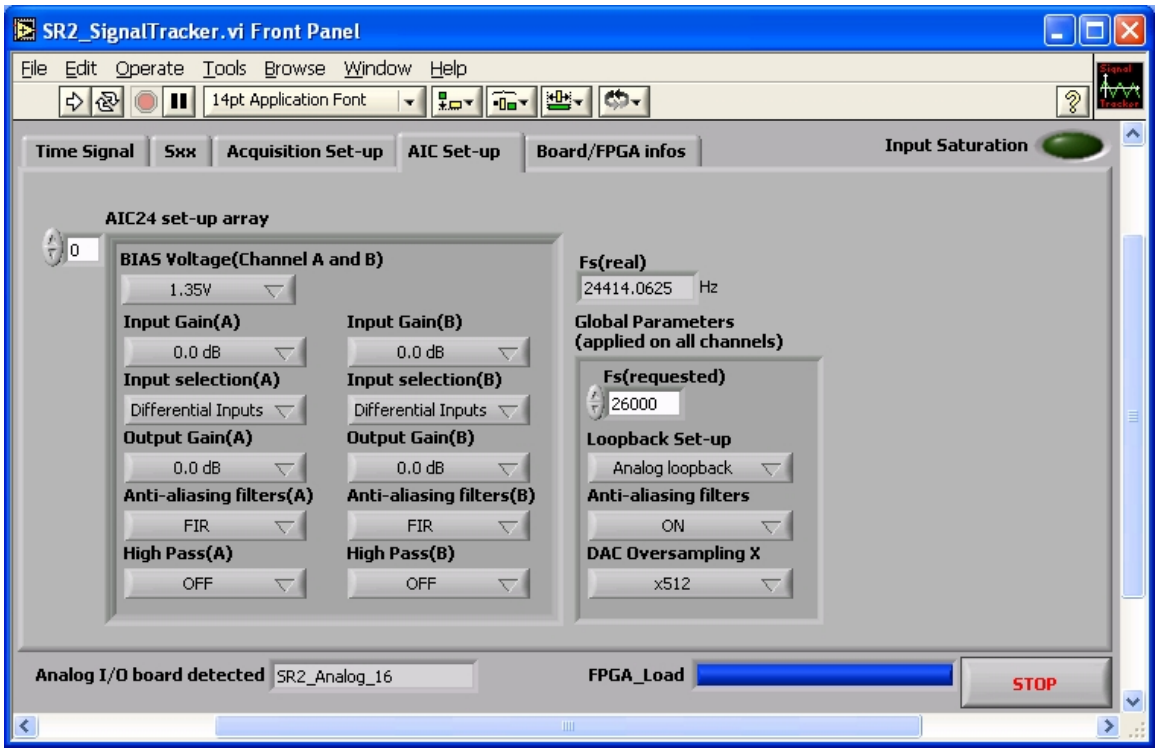


Figure 6 : Application SR2_SignalTracker - Onglet Configuration AIC

Contrôle AIC_Setup_Array

Ce tableau comprend un champ de configuration pour chaque puce AIC (et non pour chaque canal). Par défaut, c'est le champ de l'AIC 0 qui est affiché. Pour accéder à d'autres AIC (0 à 7), il suffit de modifier le numéro d'index dans le coin supérieur gauche du champ. Le champ comprend une configuration pour la tension de polarisation (qui est commune aux deux canaux de l'AIC), ainsi qu'un ensemble de commandes pour chaque canal (A ou B).

Contrôle des paramètres globaux

Ce champ comprend tous les ajustements communs à tous les AIC. Le champ Fs(requested) contient la fréquence d'échantillonnage demandée en Hz. L'application ajuste les différents compteurs de l'AIC pour obtenir une fréquence d'échantillonnage réelle la plus proche possible de la demande. Elle prend automatiquement en compte les différentes limitations sur les valeurs des compteurs.

Indicateur Fs(réel)

Ce champ indique la fréquence d'échantillonnage réelle qui a été atteinte, compte tenu de la résolution finie des compteurs AIC, ainsi que des diverses limitations sur les valeurs des compteurs.

Infos sur les cartes/FPGA Tab

Cet onglet présente des informations sur la révision matérielle du Signal_Ranger_mk2, le numéro d'identification du pilote et le fichier de configuration du FPGA.

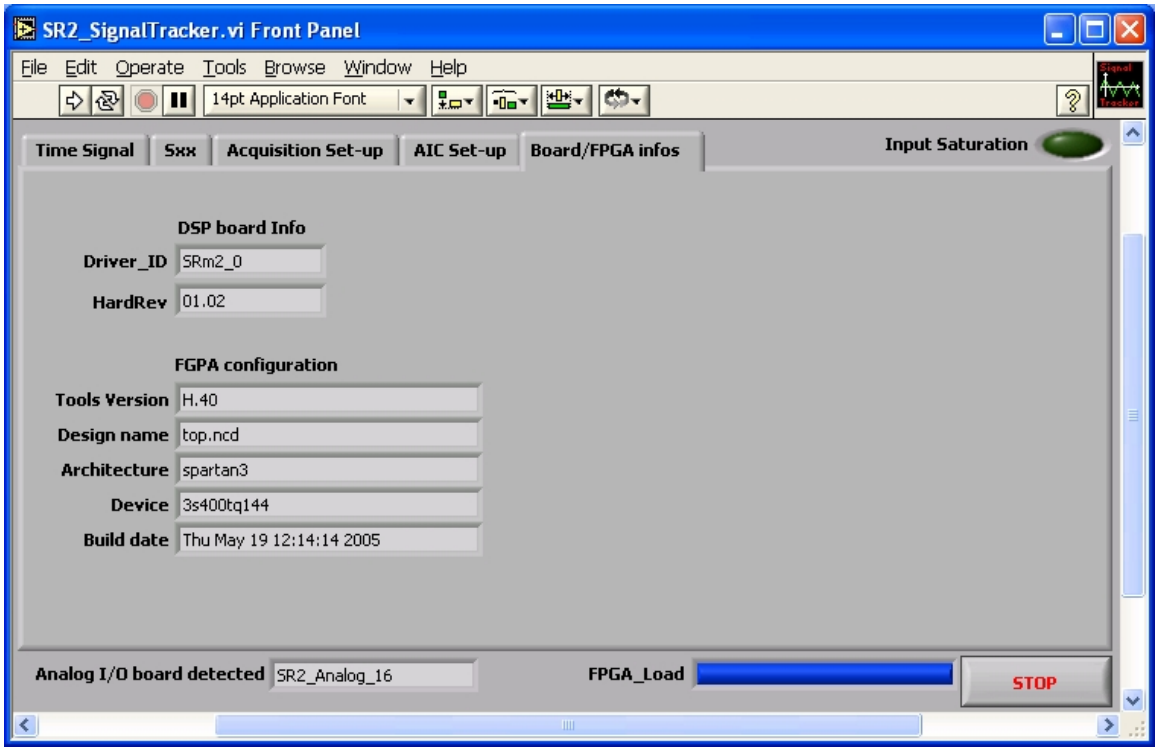


Figure 7 : Application SR2_SignalTracker - Onglet Infos carte/FPGA

Pilote AIC et code d'exemple

Vue d'ensemble

Un pilote pour les E/S analogiques (AIC) est fourni, ainsi que le code DSP d'une application de démonstration qui utilise ce pilote, ainsi qu'un projet "shell" vide. Le code source du pilote AIC se trouve dans le dossier *AIC24Driver*. Le code source de l'application de démonstration se trouve dans le dossier *SignalTracker_DSP_Code*. Ce dossier contient le code DSP de l'application de démonstration *SignalTracker* mentionnée ci-dessus. Le code source du projet shell se trouve dans le dossier *IO_Shell_DSP_Code*. Le projet shell constitue un excellent point de départ pour développer du code DSP utilisant les AIC.

Le pilote a été optimisé en langage assembleur, mais peut être utilisé soit en C, soit en langage assembleur. Il se présente sous la forme de plusieurs bibliothèques d'objets DSP *CodeDSPAIC24Driver_ii.lib*, où *ii* représente le nombre d'E/S analogiques effectivement présentes sur la carte. Le pilote contient des fonctions appelables en C pour configurer et utiliser les AIC. Une fonction appelée *dataprocess* est fournie en C, où les développeurs peuvent commodément placer leur propre code de traitement des E/S analogiques.

Le pilote utilise le DMA pour communiquer avec les AIC pour une efficacité maximale.

Configurations de la carte

La configuration de carte la plus courante contient 16 entrées/sorties analogiques (8 AIC). Cependant, il existe d'autres configurations de cartes avec un nombre réduit d'entrées/sorties analogiques. Pour chacune de ces configurations, une bibliothèque de pilotes distincte doit être utilisée. Le tableau ci-dessous énumère les configurations et le pilote applicable :

Nombre d'E/S	Numéros de canaux	Configuration	Bibliothèque des conducteurs
16	0 à 7 (McBSP0) 8 à 15 (McBSP1)	4 AIC (8 canaux) sur chaque McBSP	CodeDSPAIC24Driver_16.lib
8	0 à 3 (McBSP0) 8 à 11 (McBSP1)	2 AIC (4 canaux) sur chaque McBSP	CodeDSPAIC24Driver_8.lib
4	0 à 1 (McBSP0) 8 à 9 (McBSP1)	1 AIC (2 canaux) sur chaque McBSP	CodeDSPAIC24Driver_4.lib

Tableau 4 : Liste des configurations de cartes et des pilotes pris en charge

Lors de la liaison du code de l'application DSP, il est important de le lier à la bibliothèque de la carte concernée. Si la configuration de la carte est inconnue, l'application de démonstration *SignalTracker* ou l'application *SR2_Analog_Self_Test* peuvent être utilisées pour déterminer le nombre de canaux présents sur la carte. Par souci de simplicité, sauf indication contraire, seul le cas le plus courant de 16 canaux d'E/S est abordé ici.

Logique FPGA

Le fonctionnement du pilote suppose que la logique *SR2_Analog_16.rbt* est chargée dans le FPGA et fonctionnelle. Cette logique dirige la *sortie CLKO* du DSP vers les entrées de l'horloge maîtresse de l'AIC. Elle génère également et synchronise les entrées de réinitialisation et de mise hors tension de l'AIC avec l'horloge maîtresse. Le pilote ne charge pas cette logique FPGA. La logique doit être chargée par un autre moyen (à partir de la Flash ROM à la mise sous tension, à partir de l'application PC ou dynamiquement à partir du code DSP, avant qu'une fonction du pilote ne soit appelée). Les différentes options de chargement de la logique FPGA sont documentées dans le manuel de l'utilisateur du Signal Ranger_mk2. Le fonctionnement du pilote suppose également qu'une horloge de 75 MHz est présente sur la sortie *CLKO du* DSP.

Flux de données

Après la configuration, les 8 canaux de chaque banque commencent à échanger des échantillons et des données de configuration vers/depuis le DSP, via McBSP0 et McBSP1. Chaque McBSP supporte une banque de 8 canaux (1 maître et 7 esclaves). Ces 8 canaux échantillonnent les signaux analogiques et transmettent/reçoivent les données en séquence. Les AIC communiquent avec le DSP en mode *turbo*, de sorte que les temps d'échantillonnage et la période de transmission pour chacun des huit canaux d'une banque se produisent aussi vite que possible au début de chaque période d'échantillonnage. Le temps de transmission des entrées/sorties dure 3,41 µs par canal (1 mot de données et 1 mot de configuration). Le cas le plus défavorable est celui de 8 canaux sur chaque McBSP. Dans ce cas, la période de transmission pour tous les canaux d'une banque dure 27,31 µs. La période d'échantillonnage la plus courte est de 38,46 µs (26 kHz). Pour cette période d'échantillonnage, il y a 11,15 µs d'inactivité après la période de transmission, avant le début de la période de transmission suivante.

L'échange de données d'échantillonnage et de configuration de/vers chaque AIC, de/vers un tampon en mémoire, est effectué en arrière-plan par quatre canaux DMA (canaux DMA 0 à 3). À chaque période d'échantillonnage, une fois que toutes les données d'échantillonnage et de configuration de l'AIC ont été échangées, l'interruption du canal DMA 0 (DMAC0) est déclenchée. Cette routine de service d'interruption remet en mémoire tampon les données d'entrée et de sortie dans une structure accessible à l'utilisateur (*IOBuf)* et appelle la fonction de traitement des signaux définie par l'utilisateur (*dataprocess*), qui peut être écrite en C. La fonction *dataprocess* définie par l'utilisateur est présentée sous la forme d'un TRAP (interruption logicielle n° 30). *Dataprocess* récupère normalement les échantillons d'entrée dans la structure *IOBuf* et prépare (écrit) les prochains échantillons de sortie dans la structure *IOBuf*. Il dispose pour cela d'une période d'échantillonnage complète.

Après que le drapeau DMAC0 passe à un, l'interruption ne doit pas être maintenue pendant une longue période, en particulier à des périodes d'échantillonnage élevées, sinon des échantillons sont perdus. Pour 16 canaux, la période de transmission dure 27,31 µs, donc à la fréquence d'échantillonnage maximale de 26 kHz (période d'échantillonnage de 38,46 µs), le temps de latence maximal pour l'interruption DMAC0 est de 38,46 µs - 27,31 µs = 11,15 µs.

Étant donné que les données d'entrée/sortie AIC sont remises en mémoire tampon, le *processus de données de la* fonction définie par l'utilisateur a la garantie que les données ne changeront pas jusqu'au déclenchement de l'interruption DMAC0 suivante. Cela signifie que la fonction de *traitement des données* définie par l'utilisateur peut être exécutée pendant toute la durée de la période d'échantillonnage (38,46 µs à la fréquence d'échantillonnage maximale de 26 kHz). Elle doit cependant être terminée avant le prochain déclenchement de l'interruption DMAC0, sinon une période d'échantillonnage sera perdue et les mêmes échantillons de sortie seront envoyés aux AIC pendant deux périodes consécutives.
Étant donné qu'elle est appelée en tant que piège à partir de la routine de service d'interruption DMA, la fonction de *traitement des données* n'est normalement pas interruptible. Elle peut cependant être rendue interruptible en remettant le bit INTM à 0.

À chaque période d'échantillonnage, le DSP échange des échantillons et des données de configuration avec chaque AIC. La structure *IOBuf*, accessible à l'utilisateur, contient les échantillons de sortie ainsi que les mots de configuration à envoyer aux AIC. Normalement, après la configuration, il n'est pas nécessaire d'envoyer d'autres mots de configuration aux canaux AIC. Par conséquent, la fonction *start_aic24* qui configure les AIC écrit des mots de configuration initialisés à zéro dans les membres de configuration de la structure *IOBuf*. Le code utilisateur qui n'a pas besoin de reconfigurer les AIC de manière dynamique ne doit tout simplement pas écrire ces membres de configuration. Si une reconfiguration dynamique est nécessaire, le code DSP de l'utilisateur doit écrire les codes de configuration appropriés dans le membre de configuration de la structure *IOBuf* dans la fonction *dataprocess*. Il ne doit pas manquer d'écrire les codes de configuration initialisés à zéro lors du prochain appel de la fonction *dataprocess*. Dans le cas contraire, les codes de reconfiguration dynamique seront envoyés en continu aux AIC.

Structures et fonctions accessibles à l'utilisateur

La bibliothèque *CodeDSPAIC24Driver_ii.lib* définit et alloue les structures suivantes accessibles à l'utilisateur :

aic24reg

Cette structure est conçue pour contenir les valeurs des registres de configuration pour un maximum de 16 canaux. La structure doit être initialisée avec les paramètres souhaités avant l'exécution de la fonction *start_aic24*. La structure est définie dans le fichier d'en-tête *AIC24Driver.h.*

La structure est définie comme suit :

```
struct aicreg_rec {
                                int  pReg4_m[16]  ;
                                int  pReg4_n[16]  ;
                                int    Reg3A [16] ;
                                int    Reg3B [16] ;
                                int    Reg3C [16] ;
                                int    Reg4_m [16] ;
                                int    Reg4_n [16] ;
                                 int    Reg5A [16] ;
                                 int    Reg5B [16] ;
                                 int    Reg5C [16] ;
                                 int    Reg6A [16] ;
                                 int    Reg6B [16] ;
                                int    Reg1    [16] ;
                                int    Reg2    [16] ;
                                } ;
```

La structure est réservée par la bibliothèque du pilote comme suit :

```
struct aicreg_rec              aic24reg ;
```

Remarque : La structure est réservée automatiquement à la suite de l'inclusion de la bibliothèque de pilotes dans le code DSP utilisateur. Il n'est pas nécessaire que le code DSP utilisateur réserve cette structure.

La même structure est définie et utilisée pour toutes les configurations de la carte. Cependant, seuls les registres de configuration des canaux AIC existants doivent être initialisés. Le tableau suivant énumère les registres de configuration utilisés pour chaque configuration de carte. La notation générique "reg_x" est utilisée dans le tableau 5 pour représenter tout registre de configuration AIC spécifique. La disposition décrite dans le tableau 5 est valable pour tous les registres de configuration AIC.

Configuration de la carte	Nombre total de canaux	Registres de configuration
4 AIC (8 canaux) sur chaque McBSP	16	reg_x[0] à reg_x[15]
2 AIC (4 canaux) sur chaque McBSP	8	reg_x [0] à reg_x [3] (McBSP0) reg_x [8] à reg_x [11] (McBSP1)
1 AIC (2 canaux) sur chaque McBSP	4	reg_x [0] to reg_x [1] (McBSP0) reg_x [8] à reg_x [9] (McBSP1)

Tableau 5 : Registres de configuration utilisés en fonction de la configuration de la carte

Remarque : Tous les symboles doivent avoir un préfixe "_" lorsqu'ils sont utilisés à partir du code assembleur. Par exemple, aic24reg devient _aic24reg en code assembleur.

iobuf

Cette structure est conçue pour contenir les échantillons d'entrée et de sortie vers/depuis les AIC, ainsi que les mots de configuration à envoyer aux AIC dans le cas d'une reconfiguration dynamique. Le code utilisateur lit les registres *min[i]* et écrit les registres *mout[i]* à chaque appel de la fonction *dataprocess*. Normalement, il n'est pas nécessaire de reconfigurer dynamiquement les AIC, de sorte que seuls les échantillons de sortie sont mis à jour dans le tableau *mout[i].* Les mots de configuration sont initialisés à zéro par la fonction *start_aic24* et ne sont normalement jamais modifiés. Ils n'ont donc aucun effet sur la configuration.
Le code DSP de l'utilisateur dispose d'une période d'échantillonnage complète pour exécuter la fonction de *traitement des données.* Si la fonction n'est pas terminée au cours d'une période d'échantillonnage, les échantillons d'entrée sont écrasés par les nouveaux échantillons et les mêmes échantillons de sortie sont envoyés aux AIC.
La structure est définie dans le fichier d'en-tête *AIC24Driver.h.*

La structure est définie comme suit :

```
struct       iobuf_rec          {
                                int    min[32] ;
                                int    mout[32] ;
                                } ;
```

La structure est réservée par la bibliothèque du pilote comme suit :

```
struct       iobuf_rec          iobuf ;
```

Remarque : La structure est réservée automatiquement à la suite de l'inclusion de la bibliothèque de pilotes dans le code DSP utilisateur. Il n'est pas nécessaire que le code DSP utilisateur réserve cette structure.

Les registres d'entrée et de sortie contiennent d'abord les données d'échantillonnage et ensuite les données de configuration. La même structure est définie et utilisée pour toutes les configurations de la carte. Cependant, seuls les registres de données des canaux AIC existants doivent être utilisés. Le tableau suivant énumère les registres de données utilisés pour chaque configuration de carte.

Configuration de la carte	Nombre total de canaux	Tampon d'entrée/sortie
4 AIC (8 canaux) sur chaque McBSP	16	min[0] à min[7] : échantillons d'entrée pour les canaux 0 à 7 min[8] à min[15] : mots de configuration des entrées pour les canaux 0 à 7 min[16] à min[23] : échantillons d'entrée pour les canaux 8 à 15 min[24] à min[31] : mots de configuration d'entrée pour les canaux 8 à 15 mout[0] à mout[7] : échantillons de sortie pour les canaux 0 à 7 mout[8] à mout[15] : mots de configuration de sortie pour les canaux 0 à 7 mout[16] à mout[23] : échantillons de sortie pour les canaux 8 à 15 mout[24] à mout[31] : mots de configuration de sortie pour les canaux 8 à 15 canaux 8 à 15
2 AIC (4 canaux) sur chaque McBSP	8	min[0] à min[3] : échantillons d'entrée pour les canaux 0 à 3 min[4] à min[7] : mots de configuration des entrées pour les canaux 0 à 3 min[8] à min[15] : non utilisé min[16] à min[19] : échantillons d'entrée pour les canaux 8 à 11 min[20] à min[23] : mots de configuration d'entrée pour les canaux 8 à 11 min[24] à min[31] : non utilisé mout[0] à mout[3] : échantillons de sortie pour les canaux 0 à 3 mout[4] à mout[7] : mots de configuration de sortie pour les canaux 0 à 3 mout[8] à mout[15] : non utilisé mout[16] à mout[19] : échantillons de sortie pour les canaux 8 à 11 mout[20] à mout[23] : mots de configuration de sortie pour les canaux 8 à 11 mout[24] à mout[31] : non utilisé
1 AIC (2 canaux) sur chaque McBSP	4	min[0] à min[1] : échantillons d'entrée pour les canaux 0 à 1 min[2] à min[3] : mots de configuration des entrées pour les canaux 0 à 1 min[4] à min[15] : non utilisé min[16] à min[17] : échantillons d'entrée pour les canaux 8 à 9 min[18] à min[19] : mots de configuration d'entrée pour les canaux 8 à 9 min[20] à min[31] : non utilisé mout[0] à mout[1] : échantillons de sortie pour les canaux 0 à 1 mout[2] à mout[3] : mots de configuration de sortie pour les canaux 0 à 1 mout[4] à mout[15] : non utilisé mout[16] à mout[17] : échantillons de sortie pour les canaux 8 à 9 mout[18] à mout[19] : mots de configuration de sortie pour les canaux 8 à 9 mout[20] à mout[31] : non utilisé

Tableau 6 : Registres d'E/S utilisés en fonction de la configuration de la carte

Remarque : Tous les symboles doivent avoir un préfixe "_" lorsqu'ils sont utilisés à partir du code assembleur. Par exemple, iobuf devient _iobuf en code assembleur.

start_aic24

Cette fonction configure les AIC et lance le processus de conversion des AIC. Elle n'a pas d'arguments. Elle utilise les valeurs de configuration des registres trouvées dans la structure aic24reg et configure les AIC en conséquence. Ces valeurs doivent être initialisées avant d'appeler *start_aic24*. Elles définissent les paramètres de l'AIC tels que la fréquence d'échantillonnage, le gain d'entrée et de sortie, etc. selon les fonctions des registres de l'AIC. Après l'exécution de cette fonction, la fonction de traitement définie par l'utilisateur, appelée *dataprocess*, commence à être déclenchée à chaque période d'échantillonnage. La bibliothèque d'interface AIC LabVIEW comprend un VI pour générer automatiquement le contenu des registres en fonction des fonctions AIC sélectionnées par l'utilisateur.
La fonction est définie dans le fichier d'en-tête *AIC24Driver.h.*

Remarque : Tous les symboles doivent avoir un préfixe "_" lorsqu'ils sont utilisés à partir du code assembleur. start_aic24 doit écrit _start_aic24 dans le code assembleur.

stop_aic24

Cette fonction arrête le processus de conversion AIC. Après l'exécution de cette fonction, la fonction de *traitement des données* définie par l'utilisateur cesse d'être déclenchée et tous les AIC émettent continuellement des échantillons zéro.
La fonction est définie dans le fichier d'en-tête *AIC24Driver.h.*

Remarque : Tous les symboles doivent avoir un préfixe "_" lorsqu'ils sont utilisés à partir du code assembleur. stop_aic24 doit écrit _stop_aic24 dans le code assembleur.

processus de données

Cette fonction est déclarée par la bibliothèque du pilote AIC, mais doit être fournie par l'utilisateur. Elle contient généralement le code DSP qui lit les échantillons d'entrée dans la structure iobuf, effectue le traitement du signal entre les entrées et les sorties et écrit les échantillons de sortie dans la structure iobuf. Notez que la fonction *dataprocess* est en fait un TRAP.
En assembleur, cette fonction doit protéger tous les registres qu'elle utilise et doit être terminée par une instruction RETI. Le symbole *dataprocess* doit être déclaré à l'aide de la directive *.global.*
En C, cette fonction doit être définie avec le mot-clé *interrupt.*

Remarque : Tous les symboles doivent avoir un préfixe "_" lorsqu'ils sont utilisés à partir du code assembleur. dataprocess doit être écrit _dataprocess en assembleur.

Ressources utilisées

Le pilote AIC utilise les ressources matérielles suivantes :

- McBSP0 et McBSP1 sont tous deux utilisés par le pilote.
- Les canaux DMA 0 à 3 sont utilisés par le pilote.
- L'interruption DMAC0 est utilisée par le pilote.
- Le TRAP #30 est utilisé par le conducteur.
- Le pilote utilise 1045 octets pour le code et 352 mots pour les données.
- La sortie DSP *CLKO* est utilisée pour émettre l'horloge maîtresse de l'AIC à 75 MHz.

Restrictions

Les restrictions suivantes s'appliquent au développement de code C ou de code assembleur à l'aide du pilote AIC :

- La fonction de traitement des E/S définie par l'utilisateur, *dataprocess*, doit être présente dans le code de l'utilisateur. Si la fonction est définie en C, elle doit être définie avec le spécificateur d'*interruption*. De cette manière, le compilateur effectue une sauvegarde complète du contexte lors de l'entrée dans la fonction. Si elle est définie en assembleur, tous les registres DSP utilisés dans la fonction doivent être protégés à l'entrée et restaurés avant que la fonction ne soit activée.

retour. Il doit se terminer par l'instruction RETI. Le symbole du *processus de données* doit être défini à l'aide de la directive *.global.*

- Pour que le DMA fonctionne correctement, la fréquence d'échantillonnage doit être identique pour tous les AIC des deux banques.
- Tous les symboles et étiquettes accessibles en C définis dans la bibliothèque *CodeDSPAIC24Driver_ii.lib* doivent avoir un préfixe "_" lorsqu'ils sont utilisés en langage assembleur.
- L'ensemble des sections de code et de données (y compris la section .bss) du *fichier CodeDSPAIC24Driver_ii.lib*
  doit être liée dans la mémoire interne.
- L'interruption logicielle n° 30 est utilisée par le pilote pour appeler la fonction utilisateur du *processus de données.* Le vecteur de cette interruption logicielle doit être correctement déclaré. Voir le code DSP de la démo *SignalTracker* pour un exemple de *vectors.asm* correct.

Enfermement du conducteur

Étant donné que le pilote s'appuie sur le DMA pour les transferts d'échantillons vers et depuis les AIC, certaines situations peuvent entraîner un blocage. La DARAM ne peut supporter que 2 accès par cycle, pour lesquels le CPU a toujours la priorité sur les accès DMA et HPI. Certaines séquences d'instructions rares, lorsqu'elles sont exécutées dans une boucle serrée, nécessitent ces deux accès à la DARAM par cycle, ce qui empêche complètement l'accès au DMA du bloc DARAM à partir duquel le code est exécuté. Cette condition ne se produit que lorsque le code est exécuté à partir du même bloc DARAM que celui où se trouve le tampon DMA. Dans ces conditions, les transferts DMA ne sont jamais terminés. Par conséquent, l'interruption #30 n'est jamais déclenchée et la fonction utilisateur *dataprocess* n'est jamais appelée. Cette situation peut s'accompagner d'un blocage USB si le code s'exécute à partir du premier bloc DARAM, car le HPI utilise le DMA pour les transferts et est soumis aux mêmes règles de priorité (voir Signal_Ranger_mk2_UserManual.pdf).
Plusieurs solutions de contournement peuvent être appliquées :

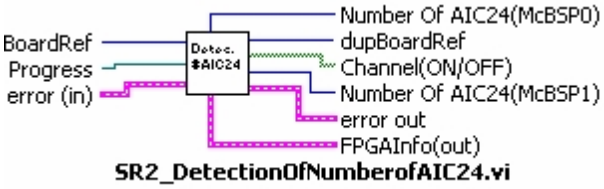
- Le tampon DMA peut être déplacé vers un autre bloc DARAM que le bloc contenant boucle de code incriminée.
- La boucle de code incriminée peut être modifiée pour éviter de nécessiter deux accès par cycle. Une solution simple consiste généralement à insérer des instructions NOP dans la boucle. Cette situation étant très rare, la plupart des réorganisations de code résoudront généralement le problème.

Bibliothèque LabVIEW de l'AIC

Une bibliothèque LabVIEW est fournie pour gérer les AIC. Cette bibliothèque s'appuie sur divers codes DSP exécutables pour fournir les fonctions. Les codes DSP sont chargés selon les besoins des fonctions de la bibliothèque.

SR2_DetectionOfNumberofAIC24.vi

Ce VI détecte le nombre d'AIC présents dans chaque banque McBSP. Le VI s'appuie sur la logique FPGA *SR2_Analog_16.rbt*, ainsi que sur le code DSP *SR2_AIC24Detection.out.* Les deux sont chargés dynamiquement par le VI. Tout code DSP en cours d'exécution est interrompu. Toute logique FPGA précédemment implémentée est effacée.



Contrôles :

- **BoardRef** : Il s'agit d'un numéro qui pointe vers l'entrée correspondant à la carte dans la base de données du
  Structure d'information de la carte globale. Elle est créée par SR2_Base_Open_Next_Avail_Board.vi
- **Progression** Il s'agit d'une référence qui peut être connectée à une barre de progression. La barre de progression varie au cours de l'exécution du VI entre 0 (le processus vient de commencer) et 1 (le processus est terminé).

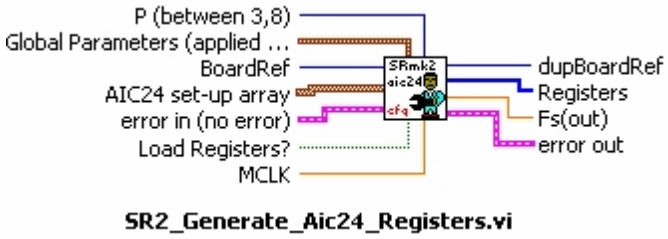
- **Erreur dans** : Grappe d'erreurs de type instrument LabView. Contient le numéro d'erreur et la description de l'erreur précédente.

Indicateurs :

- **DupBoardRef** : Il s'agit d'un numéro qui pointe vers l'entrée correspondant à la carte dans la *structure d'information globale de la carte*. Il est créé par SR2_Base_Open_Next_Avail_Board.vi Utilisez cette sortie pour propager le numéro de référence à d'autres Vis.
- **Sortie d'erreur** : Grappe d'erreurs de type instrument LabView. Contient le numéro et la description de l'erreur.
- **FPGAInfo(out)** Groupe d'informations spécifiques à la logique FPGA chargée.
- **Nombre d'AIC24(McBSP0)** Nombre d'AIC découverts dans la chaîne McBSP0.
- **Nombre d'AIC24(McBSP1)** Nombre d'AICs découverts dans la chaîne McBSP1.
- **Canal(ON/OFF)** Il s'agit d'un tableau booléen à 16 éléments. Chaque élément est VRAI lorsque l'AIC correspondant est présent, et FAUX lorsque l'AIC est absent.

SR2_Generate_AIC24_Registers.vi

Ce VI génère les registres de configuration pour les fonctions AIC spécifiées par l'utilisateur et les charge éventuellement dans la structure *aic24reg*. Lorsque *Load Register* est faux, ce VI ne nécessite aucun code DSP ou logique FPGA. Le contenu des registres est fourni à l'utilisateur à titre d'information. Lorsque *Load Register* est vrai, la logique FPGA *SR2_Analog_16.rbt* doit être chargée dans le FPGA, et un code DSP utilisant *CodeDSPAIC24Driver_ii.lib* doit être chargé dans la mémoire DSP. Le code DSP n'a pas besoin d'être en cours d'exécution lorsque la structure *aic24reg* est initialisée. Tout code DSP implémentant le *CodeDSPAIC24Driver_ii.lib* est compatible avec le VI.



Contrôles :

- **BoardRef** : Il s'agit d'un numéro qui pointe vers l'entrée correspondant à la carte dans la base de données du
  Structure d'information de la carte globale. Elle est créée par SR2_Base_Open_Next_Avail_Board.vi
- **Erreur dans** : Grappe d'erreurs de type instrument LabView. Contient le numéro d'erreur et la description de l'erreur précédente.
- **P** Cette commande définit la valeur du diviseur P dans les registres de l'AIC. En raison d'un bogue de silicium sur cet AIC, cette commande doit être laissée déconnectée ou explicitement réglée sur 8. Les AIC fabriqués après juillet 2005 ne présentent pas ce bogue.
- **Paramètres globaux** Il s'agit d'un groupe qui contient tous les paramètres qui agissent sur tous les AIC de la carte.
- **Tableau de configuration de l'AIC24** Ce tableau de clusters contient tous les paramètres spécifiques à chaque AIC et/ou à chaque canal. Ce tableau doit toujours contenir 8 éléments (un par AIC), même pour une carte qui a moins de 8 AIC. Les éléments sont toujours numérotés de 0 à 3 pour la banque McBSP0, et de 4 à 7 pour la banque McBSP1. Par conséquent, pour une carte qui n'a que 2 AIC dans chaque banque, les éléments de configuration pertinents seraient 0 et 1, et 4 et 5.
- **Charger les registres** Ce booléen détermine si le contenu des registres AIC est chargé ou non dans la structure de configuration *aic24reg*. Notez que le chargement de la structure *aic24reg* en lui-même ne reconfigure pas les AIC. Pour reconfigurer les AIC, la fonction DSP *start_aic24* doit être appelée après le chargement. Pour que le chargement puisse avoir lieu, la mémoire DSP doit être chargée avec un code qui met en œuvre le *CodeDSPAIC24Driver_ii.lib*. Ce code DSP ne doit pas être en cours d'exécution au moment du chargement.

- **MCLK** Règle la fréquence de l'horloge maîtresse. Cette fréquence est de 75 MHz par défaut. C'est la fréquence par défaut générée par la sortie DSP ClkOut, et qui est nécessaire pour piloter les AIC. En général, cette commande ne doit pas être connectée. Cependant, si l'utilisateur a besoin de piloter les AIC en utilisant une fréquence MClk différente (par exemple en utilisant les DCM dans le FPGA), cette entrée peut être utilisée pour supporter cette autre fréquence.

Indicateurs :

- **DupBoardRef** : Il s'agit d'un numéro qui pointe vers l'entrée correspondant à la carte dans la *structure d'information globale de la carte*. Il est créé par SR2_Base_Open_Next_Avail_Board.vi Utilisez cette sortie pour propager le numéro de référence à d'autres Vis.
- **Sortie d'erreur** : Grappe d'erreurs de type instrument LabView. Contient le numéro et la description de l'erreur.
- **Fs(out)** Il s'agit de la fréquence d'échantillonnage exacte qui est réalisable compte tenu des compteurs de précision finie dans les AIC, ainsi que de diverses contraintes sur ces compteurs. Le VI essaie de trouver la configuration de registre qui produira la fréquence d'échantillonnage la plus proche de la demande de l'utilisateur.
- **Registres** Ce tableau contient tous les contenus des registres permettant d'obtenir la configuration demandée. Si *Load Registers* est VRAI, le contenu de ces registres est automatiquement chargé dans la structure *aic24reg* de la mémoire du DSP.